AN IMPROVED METHOD OF PHYSICAL INTERACTION AND SIGNAL FLOW MODELING FOR SYSTEMS ENGINEERING

Mehdi Dadfarnia\textsuperscript{a}
Conrad Bock\textsuperscript{a}
Raphael Barbau\textsuperscript{b}

\textsuperscript{a}National Institute of Standards & Technology
\textsuperscript{b}Engisis, LLC

2/7/2017
TABLE OF CONTENTS

- Introduction
- Method & Example
- Translation
- Conclusion
TABLE OF CONTENTS

- Introduction
- Method & Example
- Translation
- Conclusion
Requirements

- Accelerate at 4 m/s²

Analysis and testing

Designs

- 100 kw
- Hydraulic pressure
- Mechanical pressure

Systems Engineering
Requirements

Accelerate at
of 4 m/s²

Designs

hydraulic pressure

100 kw

Mechanical
engineers

Systems
engineers

Hydraulic
engineers

Prototype
engineers

Simulation
engineers

Requirements

Materials
engineers

Production
engineers

Control
engineers

Analysis
and testing
SYSTEMS MODELING LANGUAGE (SYSML)

- Most widely used graphical modeling language for systems engineering
  - International standard since 2007
  - Currently at revision 1.5
- Diagrams for:
  - Requirements, component breakdown and interconnection, behavior, parametrics
Systems engineers and domain engineers:

- Build (separate) models of physical and informational behaviors
- Exchange models with each other
- Overlapping and inconsistent system specifications in multiple languages
A MORE SPECIFIC PROBLEM

- Domain engineers often use physical interaction and signal flow analysis
- Aka lumped parameter, 1-dimensional, or network analysis
- Specifies breakdown of components into subcomponents
- Physical substances and information exchanged across component interconnections
- Lacks support for more detailed physical modeling
MODELING WITH EQUATIONS: UNIDIRECTIONALLY (SEQUENTIALLY)

- Sequential equation-based modeling
  - Lacks any structure
  - Physics obscured

- Unidirectional component modeling:
  - Lacks system structure
  - Physics still obscured

\[
\omega = \alpha(gear) \cdot v;
\]
\[
torque = u \cdot T_m \cdot \left( 1 - \beta \cdot \left( \omega/wm - 1 \right)^2 \right);
\]
\[
F = \alpha(gear) \cdot \text{torque};
\]
\[
F_r = m \cdot g \cdot Cr;
\]
\[
F_a = 0.5 \cdot \rho \cdot C_d \cdot A \cdot v^2;
\]
\[
F_g = m \cdot g \cdot \sin(\theta);
\]
\[
F_d = F_r + F_a + F_g;
\]
\[
dv = \frac{(F - F_d)}{m};
\]

• Flow rate of substance is conserved (bidirectional)

• Substance’s potential to flow is unchanged

• Flow rate x potential to flow = energy rate

\[ FR_1 + FR_2 + FR_3 = 0 \]

\[ PF_1 = PF_2 = PF_3 \]
Lacks methods for aligning with systems engineering
- Omit physical substances
- Components interact by sharing variable values for energy exchange
- Processes within components not addressed
Result of differences between modeling systems engineering structures and equation-based behaviors:

- Produces conflicting or erroneous models from misaligned structure/behavior
- Discourages communication between engineers
- Requires additional work to resolve differences
BRIDGING THE DIFFERENCE: FIND OVERLAPS AND DIFFERENCES

<table>
<thead>
<tr>
<th>Domain</th>
<th>What is flowing</th>
<th>Flow rate</th>
<th>Potential to flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical</td>
<td>Charge</td>
<td>Current</td>
<td>Voltage</td>
</tr>
<tr>
<td>Mechanics, translational</td>
<td>Linear momentum</td>
<td>Force</td>
<td>Linear velocity</td>
</tr>
<tr>
<td>Mechanics, angular</td>
<td>Angular momentum</td>
<td>Torque</td>
<td>Angular velocity</td>
</tr>
<tr>
<td>Hydraulics</td>
<td>Volume</td>
<td>Volumetric rate</td>
<td>Pressure</td>
</tr>
<tr>
<td>Thermal</td>
<td>Entropy</td>
<td>Entropy flow rate</td>
<td>Temperature</td>
</tr>
</tbody>
</table>
1. Interchange between systems tools and physical interaction and signal flow simulation tools
   a. Extension of SysML.
   b. Translation between extended SysML and simulation models.

2. Improved method for developing models:
   - Understand physical principles
   - Seek conserved physical substances and numeric information
   - Model physical interaction and signal flow within system structure
SYSML FOR SIMULATION: REUSE EQUIVALENT CONCEPTS

<table>
<thead>
<tr>
<th>Simulation Concepts</th>
<th>SysML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Models</td>
<td>Blocks with internal block diagram, but not a component of other blocks</td>
</tr>
<tr>
<td>Components</td>
<td></td>
</tr>
<tr>
<td>Atomic</td>
<td>Blocks without internal block diagram</td>
</tr>
<tr>
<td>Subsystems</td>
<td>Blocks with internal block diagram</td>
</tr>
<tr>
<td>Links</td>
<td>Connectors</td>
</tr>
<tr>
<td>Ports</td>
<td>Ports with flow properties</td>
</tr>
<tr>
<td>Equations</td>
<td>Constraint blocks</td>
</tr>
</tbody>
</table>

Almost all concepts in simulation are available in SysML
SYSML FOR SIMULATION: EXTEND FOR MISSING CONCEPTS

pkg SysML Simulation

«metaclass»
UML::Property

«stereotype»
SimConstant

{ extends a numeric or boolean value property }

«stereotype»
SimVariable

isContinuous : Boolean = true
isConserved : Boolean = false
changeCycle : Real = 0

«stereotype»
SimProperty

referTo : FlowProperty

{ extends a numeric or boolean value property, or a property typed by a SimBlock }

«stereotype»
SimBlock

{ all properties of SimBlocks have SimVariable applied }

SYSML FOR SIMULATION: EXTEND FOR MISSING CONCEPTS
METHOD STEPS

1. Sketch the total system and its physical and informational interactions
2. Model component interconnections to support physical and informational interactions
3. Model the components used in the system structure
4. Develop equations for components involved in physical interactions and signal flows
AN EXAMPLE

- Example: Automobile cruise control system
  - Commonly used in simulation modeling methods
  - Physical interactions and signal flows
- Attempt to keep constant vehicle speed (throttle) despite disturbances
  - Gravity (road slope)
  - Tire rolling resistance
  - Air effects
STEP 1: DEVELOP AN INFORMAL SKETCH OF THE TOTAL SYSTEM COMPONENTS

- Physical interactions between and within components
  - Transmission/transformation of conserved substances in system
  - Angular momentum, linear momentum, electric charge, etc

- Information sent between components
  - Communication of information from one part of system to another
  - Numeric information, such as control signals, etc
STEP 1: INFORMAL SKETCH EXAMPLE

- Total system for a cruise controller
- Physical interactions: Solid, bidirectional arrows
- Signal flows: dashed, unidirectional arrows
STEP 2.1: IDENTIFY PHYSICAL INTERACTIONS AND SIGNAL FLOWS

- Physical interactions:
  - Vehicular driving force: Transformation between angular momentum of wheels and linear momentum of vehicle
  - Air effects: Transfer of linear momentum between vehicle and air
  - Slope of road/gravity: Transformation between linear momentum and gravitational potential energy

- Signal flows:
  - Driver instructions: Sends desired speed to controller.
  - Controller commands: Sends throttle setting to engine.
  - Wheel sensor readings: Sends angular velocity to controller
STEP 2.2: DEFINE COMPONENT INTERCONNECTIONS

- Include all components, subcomponents, and interconnections between them
  - Interconnections reflect the sketch’s physical interactions / signal flows
- Use SysML Internal Block Diagram
  - Roles (parts, ports): Played by kinds of things (blocks)
  - Connectors between roles
STEP 2: INTERNAL BLOCK DIAGRAM EXAMPLE

ibd [block] CruiseControlTotalSystem

controlledVehicle : Car
  structure
    driver : Person
    SpeedSignal
    speedController : CruiseController
      ThrottleSignal
      powerSource : Engine
        SpeedSignal
        AngularMomentum
        impeller : Wheel
            Momentum
  LinearMomentum

operatingEnvironment : Earth
  structure
    atmosphere : Air
        linearMomentum
          «connector»
          airVehicleLink : LMomPotEngTransformation

    surface : Road
        «connector»
        impellerVehicleLink : ALMomentumTransformation

    IMTG
STEP 3: DEFINE COMPONENTS USED IN SYSTEM STRUCTURE

- Specifies the kinds of things playing parts and ports in Step 2
- Shows relationships between components that can be used to connect components in the system.
- Use SysML Block Definition Diagram
STEP 3: BLOCK DEFINITION DIAGRAM EXAMPLE, SYSTEM DECOMPOSITION

<table>
<thead>
<tr>
<th>BDD Cruise Control Total System, Physical Decomposition</th>
</tr>
</thead>
</table>

- **Car**
  - **references**
    - driver: Person
  - **parts**
    - speedController: CruiseController
    - powerSource: Engine

- **CruiseControlTotalSystem**
  - **connector properties**
    - airVehicleLink: LMomentumTransfer
    - gravVehicleLink: LMomPotEngTransformation

- **controlled Vehicle**
  - **References**
    - speedController: CruiseController
    - powerSource: Engine

- **Earth**
  - **parts**
    - atmosphere: Air

- **Road**
  - **Details**
    - surface

- **Wheel**
  - **impeller**
  - **interfaceBlock**
    - ML::LMomFlowElement
      - **Momentum**

- **AMomFlowComponent**
  - **interfaceBlock**
    - ML::LMomFlowElement
      - **Momentum**

- **L Momentum Total Ground**
  - **details**
    - surface
STEP 3: BLOCK DEFINITION EXAMPLE, PHYSICAL FLOWS

**bdd Cruise Control Total System, Physical Flows**

- **AMomFlowComponent**
  - sim variables
  - radius : Length

- **ML::AMomFlowElement**
  - flow properties
  - inout aMom : AngularMomentum
  - sim properties
    - [referTo=aMom] aMF : AMomFlow

- **ML::AMomFlow**
  - sim variables
    - [isConserved] trq : Torque
    - aV : AngularVelocity

- **ML::LMomFlow**
  - sim variables
    - f : Force
    - lV : LinearVelocity

- **ML::LMomFlowElement**
  - flow properties
    - inout lMom : LinearMomentum
  - sim properties
    - [referTo=lMom] IMF : LMomFlow

- **ALMomentumTransformation**
  - «association, block»

- **L MomentumGround**
  - «port» lMTG

- **toAMFC**
  - Momentum

- **toLMFC**
  - Momentum
STEP 4: DEVELOP EQUATIONS FOR COMPONENTS

- Develop equations for components modeled in Step 3, based on interactions in Step 2
- Provides equations to generate code for simulators
- SysML Parametric Diagram
  - Bind component/association block properties to equations
  - One for every component / association block

\[
\begin{align*}
\text{par [block] Spring} \\
\text{sc : SpringConstraint} \\
\text{constraints} \\
\{ \text{der(v)} = (f - k \times x) / m \} \\
\{ \text{der(x)} = v \} \\
\{ \text{pos} = x \} \\
\text{parameters} \\
\text{u.sigsp} \quad \text{f} \quad \text{pos} \quad \text{y.sigsp} \\
\text{position} \quad \text{x} \quad \text{mass} \\
\text{springcst} \quad \text{k} \quad \text{v} \quad \text{velocity}
\end{align*}
\]
STEP 4: COMPONENT EQUATION EXAMPLE

```
par [association, block] ALMomentumTransformation

aLMTC : ALMomTransComponent
  structure
  «constraint» aLMT : ALMomTransConstrain
    constraints
    ▼av=(lv-gv)/r
    ▼trq=f*r
    ▼gf=0
    parameters
    lv ▼equal ▼av, ▲IMF.IV
    f ▼equal ▼trq, ▲IMF.f
    gv ▼equal ▼gf, ▲IMCG.IMF.IV
    gf ▼equal ▼f, ▲IMCG.IMF.f

Angular Momentum
  ▲amFC : AMomFlowComponent
    structure
    ▼radius ▼equal ▼IMTG
    ▼equal ▼radius, ▲lMF

Linear Momentum
  ▲LMFC
```

Angular Momentum

Linear Momentum

```
TABLE OF CONTENTS

- Introduction
- Method & Example
- Translation
- Conclusion
TRANSLATION: PRE-PROCESSING

- SysML structures that simulation doesn’t support
  - Association blocks
  - Flow / simulation properties on components / part
  - Nested ports
- Addressed by processing before translation.
Association blocks have their own interconnected parts

Replace association blocks with their contents

Move connectors in association block to block owning connector property.
  - Links to participant properties directly linked to participants

Move binding connectors in association blocks to a parametric diagram
PRE-PROCESSING: MOVE SIMULATION / FLOW PROPERTIES FROM COMPONENTS TO PORTS

- SysML blocks can have flow / simulation properties
  - Car inherits linear momentum property and simulation property referring to it.
- Move flow / simulation properties to port.
### TRANSLATION: BETWEEN EXTENDED SYSML AND SIMULATION PLATFORMS

<table>
<thead>
<tr>
<th>Extended SysML</th>
<th>Modelica</th>
<th>Simulink / Simscape</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocks without internal block diagrams</td>
<td>Models without connections</td>
<td>Block types / Components</td>
</tr>
<tr>
<td>Blocks with internal block diagrams</td>
<td>Models with connections</td>
<td>Systems / Components</td>
</tr>
<tr>
<td>Connectors</td>
<td>Connections</td>
<td>Lines / Connections</td>
</tr>
<tr>
<td>SimConstants</td>
<td>Parameters</td>
<td>Parameters</td>
</tr>
<tr>
<td>SimVariables (conserved)</td>
<td>Flow variables</td>
<td>NA / Balancing variables</td>
</tr>
<tr>
<td>SimVariables (not conserved)</td>
<td>Variables</td>
<td>NA / Variables</td>
</tr>
<tr>
<td>SimBlocks</td>
<td>Connectors</td>
<td>NA / Domains</td>
</tr>
<tr>
<td>Ports (physical interaction)</td>
<td>Ports (informal)</td>
<td>Ports / Nodes</td>
</tr>
<tr>
<td>Ports (signal flow)</td>
<td>Input, output variables</td>
<td>Input, output ports</td>
</tr>
<tr>
<td>ConstraintBlock usages</td>
<td>Equations</td>
<td>S-functions / Equations</td>
</tr>
</tbody>
</table>
AUTOMATED TRANSLATION

SysML Tool

SysML Model File

SysML and Simulator Extensions

SysML and Simulator Extensions

Translator Program

Simulation Language Metamodels

Modelica and Mathworks

public void generate(Class urootblock, Resource r) {
    sysmlutil = new SysMLUtil(r.getResourceSet());
    slibrary = SimulinkFactory.eINSTANCE.createSLib();
    slibrary.setName(urootblock.getName() + "Library");
    while(!toProcess.size() == 0) {
        ReferenceKey rk = toProcess.pop();
        SElement element = refx.get(rk);
        if (element instanceof SSystem) {
            SSystem ssystem2 = processSys(rk.getKey()[0]);
            slibrary.getSystem().add(ssystem2.getSubsys());
        } else if (element instanceof SFunction) {
            processSFunc(rk.getKey()[0], rk.getKey()[1]);
        }
    }
}
MODEL TRANSLATION: MODELICA

- Load SysML model files as instances in UML metamodeling repository
- Apply pre-processing to instances
- Translate & emit simulation files (different platforms)
  - Components
  - Value of simulation constants
  - Conserved variables
  - Port properties & port types
  - Inputs/Outputs

```modelica
model Car
  Person driver;
  CruiseController speedController(kI.start=30.0,kI.fixed=true, kP.start=200.0,kP.fixed=true, throttleAccRatio.start=1.0, throttleAccRatio.fixed=true);
  Engine powerSource;
  Wheel impeller;
  LMomFlowComponent simCarLMom;
  AMomFlowSim simCarAMom;
  equation
  connect(speedController.speedDriverJack, driver.simPersonSpeed);
  connect(speedController.throttleActuatorJack, powerSource.simEngineThrottleSetting);
  connect(speedController.speedSensorJack, impeller.simWheelSpeed);
  connect(impeller.WheelAMom, powerSource.engineAMom);
  connect(impeller.simWheelAMom, simCarAMom);
end Car;
```
MODEL TRANSLATION: SIMSCAPE / SIMULINK

- Load SysML model files as instances in UML metamodeling repository
- Apply pre-processing to instances
- Translate & emit simulation files (different platforms)
  - Components
  - Value of simulation constants
  - Conserved variables
  - Port properties & port types
  - Inputs/Outputs

```xml
<Block BlockType="SubSystem" Name="Person" SID="2">
  <P Name="Ports">[0,1]</P>
  <System>
    <Block BlockType="Outport" Name="simPersonSpeed" SID="3">
      <P Name="Port">1</P>
      <Block BlockType="M-S-Function" Name="pc" SID="4">
        <P Name="FunctionName">PersonConstraint</P>
        <P Name="Ports">[0,1]</P>
        <Line>
          <P Name="Src">4#out:1</P>
          <P Name="Dst">3#in:1</P>
        </Line>
      </Block>
    </System>
  </Block>
</Block>

model Person
  output Velocity simPersonSpeed;
  equation
    simPersonSpeed=f(t);
end Person;
```
MODEL TRANSLATION: DEMONSTRATION
# TABLE OF CONTENTS

- Introduction
- Method & Example
- Translation
- Conclusion
DISCUSSION OF RESULTS

- Importance of physical principles
  - Flow of conserved physical substances within components and between components

- Models emphasize transformation and transmission of conserved substances, supported by equations and variable values → better model representation of system

- Include objects in operating environment of system

- More systematic modeling method for capturing system physics
METHOD BENEFITS: MODELING

- Physical processes modeled within system structure
- Equation development guided by flow and transformation of conserved substances
- Compare to piecing together equations defined elsewhere that are not tailored to the system
METHOD BENEFITS: COMPATIBILITY

- Better alignment of systems engineering and simulation models
  - Physical processes only occur within components and between neighboring components
  - Equations developed within graphical representation of system structure

- Better communication between engineers
  - Encourages systems and simulation analysts to discuss physical processes around similar, shared models
CONCLUSION

- Modeling method to facilitate integration of systems engineering models with physical interaction and signal flow simulations
  - Focusing on underlying physics (flow and transformation of conserved substances).
- More guidance in developing equations than other methods
- Encourages collaborative development of systems engineering and simulation models
QUESTIONS?